

Displaying and Interacting with Lists

Dr. Abdelkarim Erradi
CSE@QU

Outline

1. Displaying a List
2. Interacting with a List

Displaying a List

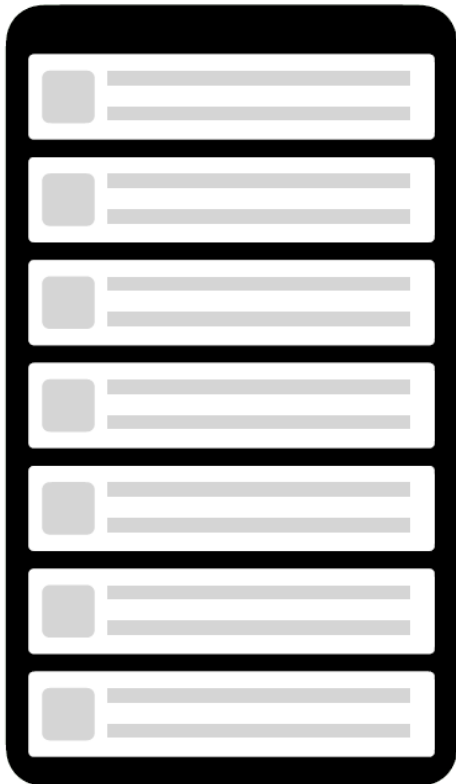
Displaying a List

- In apps it is common to display collections of items
- For displaying a small collection of items, a **Column** or **Row** layouts could be used
 - The **verticalScroll()** modifier could be applied to make the Column scrollable
 - The **horizontalScroll()** modifier could be applied to make the Row scrollable
- For displaying a large list, using a Column/Row layout can cause performance issues
 - Since all the items will be composed and laid out whether or not they are visible
 - Use a Lazy List (i.e., LazyColumn or LazyRow) to only compose and lay out items which are **visible on screen**

Displaying a List

Making the Column scrollable by using the **verticalScroll()** modifier

```
@Composable
fun SurahsList(surahs: List<Surah>) {
    Column( {modifier =
        Modifier.verticalScroll(rememberScrollState())
    })
        if (surahs.isEmpty()) {
            Text("Loading surahs failed.")
        } else {
            surahs.forEach {
                SurahCard(surah = it)
            }
        }
    }
}
```



Common Modifiers

- `Column(modifier = Modifier.verticalScroll(rememberScrollState()))`

Makes the column scrollable

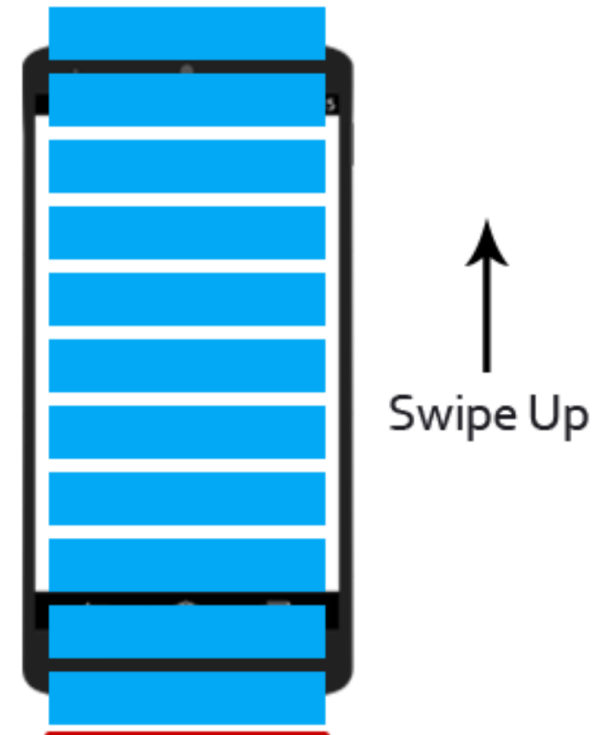
- `Row(modifier = Modifier.horizontalScroll(rememberScrollState()))`

Makes the row scrollable

- `Modifier.fillMaxWidth() /`
`.fillMaxHeight() / .fillMaxSize()` -
occupy the available space

What is a Lazy List?

- A Lazy List is a scrollable container for displaying a list of composables
 - [LazyColumn](#) produces a vertically scrolling list, and [LazyRow](#) produces a horizontally scrolling list
- A flexible container for efficiently displaying, and interacting with large sets of data
 - As user scrolls, views are created to display new items
 - Efficient as it uses a limited number of views



Lazy List methods

- Lazy List provides several functions for describing items in the layout:
 - **item()** to add a single item (e.g., header/footer)
 - **items(aList)** to add multiple items
 - **itemsIndexed(aList)** to add multiple items and provides an index

```
import androidx.compose.foundation.lazy.items
```

```
...
```

```
LazyColumn {  
    items(surahs) {  
        SurahCard(it)  
    }  
}
```



Styling a List - Content spacing

- Use [Arrangement.spacedBy\(\)](#) to add spacing in-between items

```
LazyColumn(  
    verticalArrangement = Arrangement.spacedBy(4.dp),  
) {}
```

- Similarly, for LazyRow:

```
LazyRow(  
    horizontalArrangement = Arrangement.spacedBy(4.dp),  
) {}
```

Styling a List – Content padding

- To add padding around the edges of the content pass some **PaddingValues** to the **contentPadding** parameter

```
LazyColumn(  
    contentPadding = PaddingValues(horizontal = 16.dp, vertical = 8.dp)  
) {}
```

- Adds 16.dp of padding to the horizontal edges (left and right), and then 8.dp to the vertical edges (top and bottom) of the content
- Padding is applied to the content, not to the LazyColumn itself

```

LazyColumn(contentPadding =
    PaddingValues(horizontal = 8.dp, vertical = 8.dp),
    verticalArrangement = Arrangement.spacedBy(8.dp)
) {
    item {
        Text(
            text = "سور القرآن الكريم",
            textAlign = TextAlign.Center,
            modifier = Modifier.fillMaxWidth(),
            style = TextStyle(
                fontWeight = FontWeight.Bold,
                fontSize = 24.sp,
                color = Color.Blue,
                textDirection = TextDirection.Rtl
            )
        )
    }
    items(surahs) {
        SurahCard(it)
    }
    item {
        Text(
            text = "$surahCount سورة - $ayaCount آية",
            textAlign = TextAlign.Center,
            modifier = Modifier.fillMaxWidth(),
            style = TextStyle(
                fontWeight = FontWeight.Bold,
                fontSize = 20.sp,
                color = Color.Blue,
                textDirection = TextDirection.Rtl
            )
        )
    }
}

```

Compose Lists




سور القرآن الكريم



1. الفاتحة - Al-Fatiha
Aya count: 7



2. البقرة - Al-Baqara
Aya count: 286




3. آل عمران - Aal-e-Imran
Aya count: 200




4. النساء - An-Nisa
Aya count: 176




110. النصر - An-Nasr
Aya count: 3



111. المسد - Al-Masadd
Aya count: 5



112. الإخلاص - Al-Ikhlās
Aya count: 4



113. الفلق - Al-Falaq
Aya count: 5



114. الناس - An-Nas
Aya count: 6

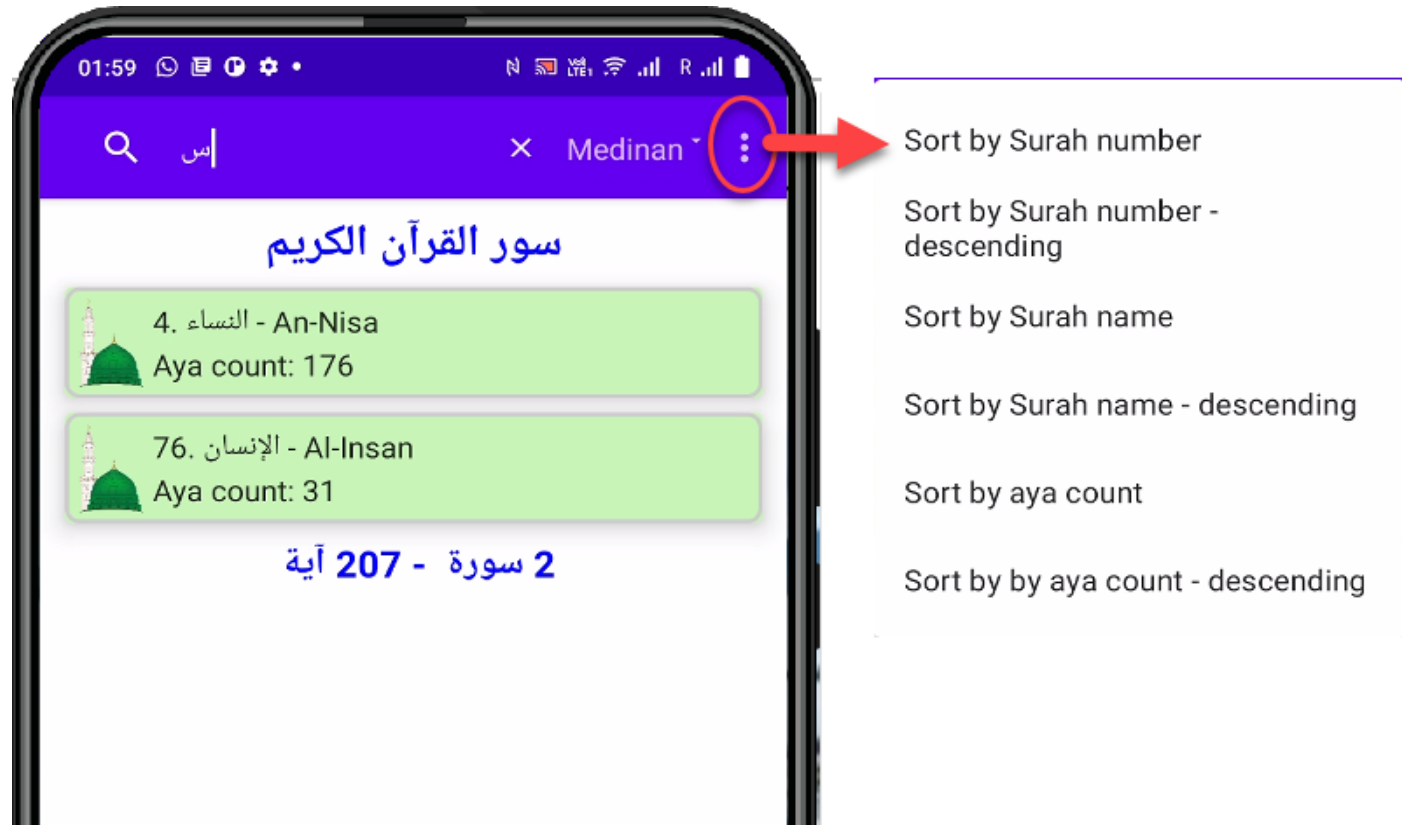
114 سورة - 6236 آية

Interacting with a List

Search

- Add a SearchBox to the top toolbar
- Handle search in the composable displaying the list

**See posted
Surah
example**



Sort

- Add sort options as menu items to the App bar menu
- Handle sort in the composable displaying the list

Summary

- Use the **verticalScroll** or **horizontalScroll** modifiers to display a small list of composables in a Column or a Row
- For dynamic and larger lists use **LazyColumn** and **LazyRow** for the vertical and horizontal scenarios, respectively
- You can program various interactions with a displayed list including *search*, *sort*, *refresh*, *add*, *update* and *delete*

Resources

- Displaying a list using Jetpack Compose

<https://developer.android.com/jetpack/compose/lists>